On the Hardware Design and Control Architecture of the Humanoid Robot Kangaroo

Adria Roig†, Sai Kishor Kothakota†, Narcis Miguel‡,
Pierre Fernbach†, Enrico Mingo Hoffman† and Luca Marchionni†

***Abstract*—Humanoid bipedal platforms are finally moving from controlled lab environments to real-world applications in unstructured scenarios. This has been recently achieved with substantial improvements in hardware and software architectures. In particular, the hardware design of bipedal humanoid robots has changed to be robust and resilient to impacts, as well as lightweight and efficient for dynamic movements. At the same time, planning and control architectures permit now to generate and control high dynamic, contact rich, motions employing the dynamics and the full body of the system.**

In this short paper, we present Kangaroo, a new humanoid bipedal robot designed by PAL Robotics for research on agile and dynamic locomotion. Kangaroo exploits novel mechanical solutions based on linear actuators and closed/parallel kinematics chains. We present the mechanical design and the whole planning and control pipeline developed to make Kangaroo jump on real experiments to demonstrate the superior robustness and resilience of the platform.

***Paper Type* – Original Work.**

I. Introduction

Boston Dynamics videos had demonstrate that humanoid robots could outperform human capabilities, by realizing fast and dynamic motions that most humans are not able to execute. To achieve those skills is necessary a perfect combination of planning and robust control software able to react against disturbances, a proper hardware design that offers enough speed and torque to perform those motions, and a reliable platform with high resilience to impacts. Following this objective, PAL Robotics has designed the new bipedal research platform Kangaroo for exploring advanced control methods for legged locomotion[1].

The robot, with 6 actuated Degrees of Freedom (DOFs) per leg, has been designed to be lightweight (12.5 kg per leg) with low moving inertia so that it is suitable for highly dynamic motions such as jumping and running. The design of Kangaroo takes advantage of closed linkage mechanisms to place all the actuators close to the torso, protecting the motors and electronics against impacts and shaping the gear ratio curve to provide high speed or high torque depending on the configuration.

Kangaroo also introduces a novel kinematic design in which the swing and the leg extension/contraction motions are completely decoupled. To jump vertically or to perform vertical motions, one single actuator per leg is activated, being highly efficient compared to many other robots in which this movement is achieved by the simultaneous activation of multiple actuators at the hip, knee, and ankle, acting in opposite directions.

All the actuators are driven with a brushless motor connected to a ball screw, providing high efficiency, low friction, and resistance to high forces in the linear direction. In the hip and ankle roll and pitch joints, linear actuators are arranged in differential configurations to increase the limb's force capability without substantially increasing reflected inertia. In addition, Kangaroo employs a new generation of in-house electronics boards for motor control and sensor acquisition, that allows embedding custom and decentralized closed-loop force/torque controllers at the actuator level. This permits running low-level controllers at a higher frequency, up to 20 kHz. The actuators are equipped with force sensors that perceive the external forces applied through the mechanism.

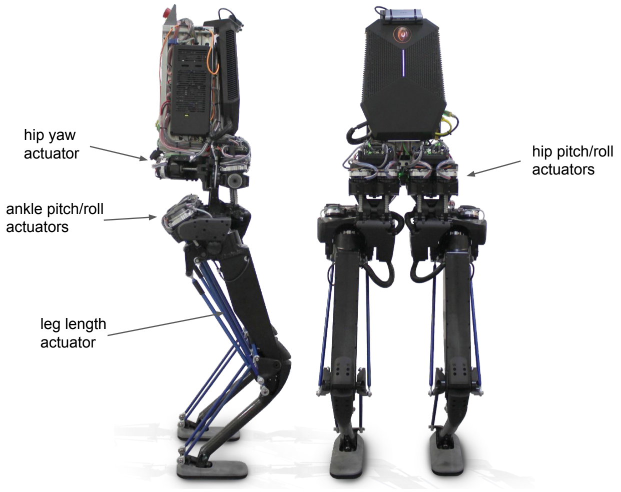


Fig. 1: Side and front view of the Kangaroo robot and actuators placing.

In terms of software, Kangaroo uses ROS as a standard framework [1] benefiting from the available functionalities for testing, visualization, and communication between processes, making it a versatile platform for researchers. ros_control [2] is in charge to deal with hardware and software abstraction layers, thereby exposing the interfaces to the controllers.

† PAL Robotics, Carrer de Pujades 77, 08005 Barcelona, Spain. E-mail: {adria.roig, sai.kishor, narcis.miguel, enrico.mingo, luca.marchionni}@pal-robotics.com

‡ Toward, Boulevard Lazare Carnot 81, 31000 Toulouse, France. E-mail: {pierre.fernbach}@toward.fr

[1] https://pal-robotics.com/robots/kangaroo/

This short paper introduces the platform, the main software architecture, and preliminary results achieved with Kangaroo.

II. Related Works

Several bipedal humanoid robots present closed linkages, in particular parallel four-bars mechanisms for the actuation of the pitch-roll ankle joints [3]–[5]. However, in this case, the actuators are moved along the shin, making the advantages of such designs localized only to the ankle part.

The Atrias robot [6] is the first of a series of famous biped systems which uses closed linkages on the legs kinematics. The new BostonDynamics Atlas [7] and AgilityRobotics Cassie/Digit [8], [9], both presents closed linkages in their mechanical design. In particular, Cassie/Digit uses electric actuators localized near its torso/pelvis with a transmission, consisting of closed-bar linkages, to transfer the motion to reduce inertia, especially during impacts while walking. Finally, RH5v2 is a recently developed, full humanoid robot, presenting closed kinematic loops both in the upper and lower body [10].

Following this new trend in the design of humanoid robots with closed and parallel kinematic linkages, this work presents the preliminary results on the modeling, planning, and control of the lower body of the Kangaroo platform.

III. The Kangaroo Robot

The Kangaroo robot has been designed using novel high-power and robust linear actuation units located near the pelvis area, while the motion is transferred to the joints through a complex system of closed and parallel linkages. Kangaroo is actuated by 12 linear ball-screw modules with electrical motors, 6 per leg, while the complete leg can be modeled by 76 DOFs, among them, 64 are passive. Figure 1 shows the location of the actuators in Kangaroo, 5 out of 6 are placed at the hip and only 1 is placed at the femur. Table I reports a comparison between linear actuators used in robotics and the 2 types of units used in Kangaroo[2].

The actuation is reported at the joint-level by a series of closed and parallel mechanisms consisting of several four-bar linkages, differential linkages, and other closed-bar linkages.

The peculiar design of these linkages on Kangaroo permits to extend the feet by maintaining constant the orientation of the ankle w.r.t. the base frame emulating a prismatic joint, and also to move the hip in the roll, pitch, and yaw direction, as well to move the ankle in the roll and pitch direction. In particular, the extension/retraction of the leg is achieved by just actuating the motor on the femur. Notice that the hip pitch and roll, as well the ankle pitch and roll are coupled, by the respective parallel linkages, forming complex nonlinear differential joints.

We developed a *reduced model* to curtail the complexity of the full model with 76 DOFs and to avoid the use of closed kinematic chains that are not fully supported by many simulators or kinematics/dynamics libraries. This reduced model of Kangaroo is used to perform Operational Space Inverse Dynamics (OSID) computation and trajectory optimization. It contains the principal links, being a good approximation for the masses and inertias in the OSID. Such a model consists of 16 DOFs, where 12 DOFs are actuated while the remaining 4 are passive. Figure 2 shows the kinematics of the reduced model. We employ a *virtual* linear joint between the hip and the ankle to emulate the effect of the transmission for the leg length, and the second open kinematic branch contains the femur and the tibia links starting at the hip and terminating at the ankle. Since this formulation contains two open chains, they need to be constrained inside the inverse dynamics (see Section V). Finally, the quantities computed using this model are then mapped to actuation space using *transmissions* which considers the full system of closed linkages (see Section IV).

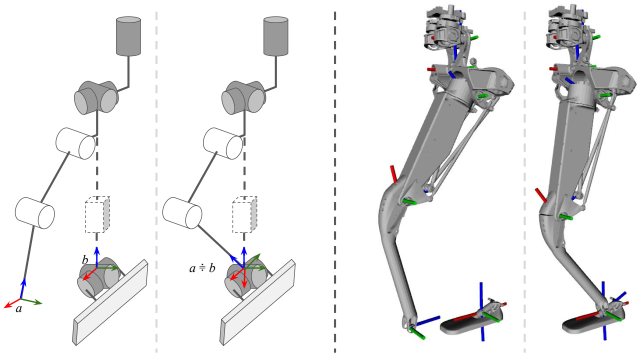


Fig. 2: On the left, the kinematics of the reduced model with the open and closed branch. On the right the reduced model visualized in RVIZ, again with open and closed branch.

IV. Actuation Mapping through Transmission

As previously written, Kangaroo control software infrastructure is based on ros_control, which provides a set of powerful libraries and tools that helps to define the structure and a properly defined functionality of each of its components. One such component in this framework is the *Transmissions*.

Transmissions interfaces are the point where the joint commands are transformed into the actuator commands, and vice versa. ros_control provides already the interface for the most common transmissions such as revolute joints, prismatic joints, or differential joints.

At this point, the transmission component of the ROS framework helps us to build the Kangaroo custom nonlinear transmission plugins based on the geometrical system of the leg where a single actuator moves multiple passive joints. This makes sure that the complexity of the closed-loop linkages is pushed down to the low-level components and thereby exposing a simplified joint structure control system, the one of the reduced model, to the high-level components such as the OSID controller.

Kangaroo has 4 different custom nonlinear transmission plugins built to control the whole leg structure:

- Virtual leg length joint,
- Ankle pitch and roll joint,

[2] Notice that the Moog-IIT ISA are based on hydraulic actuation

TABLE I: Comparison between linear actuators

	Apptronik P170 Orion [11]	**PAL**		**Moog-IIT ISA** [12]		**RH5 Knee** [13]
		LA5	**LA2**	**v2**	**v5**	
Peak Force $[kN]$	3.2	5.0	2.0	4.0	7.5	5.845
Max Speed $[\frac{m}{s}]$	0.27	0.65	0.42	-	-	0.14
Mass $[kg]$	0.786	2.1	0.78	0.920	1.6	-
Stroke $[cm]$	6.47	0.15	0.08	8.0	10.0	11.8

- Hip roll and pitch joint,
- Hip yaw joint.

One of the benefits to hide all the closed linkage complexity in the transmission component is the re-usability of previously developed controllers since most of them command the robot in joint space, and not in actuation space.

V. QP-based Operational Space Inverse Dynamics

The operational space tasks are transformed into torque references using an OSID stack solved using Quadratic Programming (QP) optimization. In addition to the classical formulation of the OSID problem, we consider as well the presence of a closed linkage in our model. Being $\mathbf{q} \in \mathbb{R}^{16} + SO(3)$ and $\boldsymbol{\nu} \in \mathbb{R}^{22}$ the robot generalized coordinates and velocities, respectively, the equations of motions can be expressed as:

$$\mathbf{M}(\mathbf{q})\dot{\boldsymbol{\nu}} + \mathbf{h}(\mathbf{q}, \boldsymbol{\nu}) = \mathbf{S}\boldsymbol{\tau} + \mathbf{J}_c^T(\mathbf{q})\mathbf{F}_c + \mathbf{J}_v^T(\mathbf{q})\boldsymbol{\lambda}, \qquad (1)$$

with $\mathbf{F}_c \in \mathbb{R}^{24}$ and $\boldsymbol{\lambda} \in \mathbb{R}^4$, constrained by

$$\mathbf{J}_c(\mathbf{q})\dot{\boldsymbol{\nu}} + \dot{\mathbf{J}}_c(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} = \mathbf{0}, \qquad (2)$$

accounting for the contacts and

$$\mathbf{J}_v(\mathbf{q})\dot{\boldsymbol{\nu}} + \dot{\mathbf{J}}_v(\mathbf{q}, \boldsymbol{\nu})\boldsymbol{\nu} = \mathbf{0}, \qquad (3)$$

accounting for the closed linkages between the hip and the ankles. Notice that $\mathbf{S}$ considers both the *underactuation* associated with the floating base *and* the open linkages in the model.

We designed a novel OSID scheme based on three main steps:

- First, a QP constrained by the first 6 rows of (1) is solved to find optimal $\dot{\boldsymbol{\nu}}^*$ and $\mathbf{F}_c^*$, considering for instance, tasks such as the Center of Mass (COM), angular momentum, base orientation and other constraints such as friction cones.
- Secondly, the Lagrange multipliers $\boldsymbol{\lambda}$ associated to the closed linkages are solved exploiting the particular structure of the reduced model[3]:

  $$\boldsymbol{\lambda}^* = \mathbf{J}_{v,u}^{-T}(\mathbf{q})\left(\mathbf{M}_{j,u}(\mathbf{q})\dot{\boldsymbol{\nu}}^* + \mathbf{h}_{j,u}(\mathbf{q}, \boldsymbol{\nu})\right), \qquad (4)$$

  where the subscripts u and j mean the *underactuated* and *joint space* part, respectively, of the matrix where the subscripts are applied.
- Third, the actuated torques $\boldsymbol{\tau}$ are retrieved substituting at the actuated part of (1) the obtained $\dot{\boldsymbol{\nu}}^*$, $\mathbf{F}_c^*$ and $\boldsymbol{\lambda}^*$.

This OSID algorithm is integrated as a ROS controller that sends computed optimal torques to a decentralized, high frequency, low-level torque controller, before sending the commands to the actuators. A further component consists of a kinematic state estimator for the pose and the velocities of the base of the robot, needed by the OSID. Finally, the references for the OSID are computed as a plugin within the controller.

This architecture allows sharing the same structure among different behaviors: stabilizing, walking, jumping, etc.

VI. Jump Planning and Stabilization

The jumping controller generates the references to make the robot jump, rectify the position and orientation in the air, and stabilize it after landing. Since the objectives and constraints involved in each phase of the jump are different they need to be replaced online.

During the first phase of the jumping, the desired height of the jump is transformed into a ramp velocity profile of the COM with a constant acceleration that will depend on the desired height. In this first stage, the classical contact constraints, i.e. friction cones and feet contacts, are active. During this phase, it is important to minimize the angular momentum to avoid undesired rotations during the next flight phase.

Once the COM reaches a desired velocity or height, the robot is considered to be in the flight phase. The flight phase consists of a parabolic shot in which the time of flight and COM height will depend on the last velocity achieved at the end of the first phase. At this point, all the contact constraints have to be deactivated and only the postural task and the angular momentum task will remain active. The angular momentum task helps to stabilize and correct possible rotations due to non-zero angular velocities at the moment of jumping.

When contacts are detected, the landing phase starts. In this phase, we add a constraint to control desired resultant ground reaction forces for each foot and the friction cone constraints and the COM tasks are activated again. The reference ground reaction forces for the landing phase are computed off-line employing trajectory-optimization based on CasADi [14]. The forces go from high, to resist the impact, to low, to actively absorb the landing, back to the nominal value to keep the height of the COM (see Figure 3). During the first and final phases, the whole motion is stabilized through a task at the

[3] In this particular case, the contact forces $\mathbf{F}_c^*$ does not appear in the computation

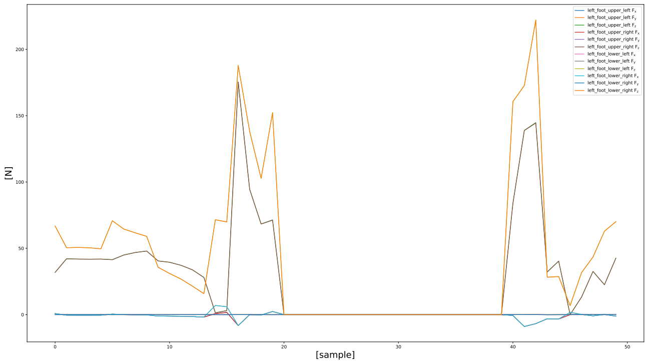


Fig. 3: Contact forces at the right foot for the jump, planned using trajectory optimization.

COM acceleration based on linear inverted pendulum model and zero moment point (ZMP):

$$\ddot{\mathbf{x}} = \omega^2 \left(\mathbf{x} - \mathbf{z}_d\right), \tag{5}$$

where $\mathbf{x}$ and $\ddot{\mathbf{x}}$ are the projected position and accelerations of the COM, respectively, $\mathbf{z}_d$ is the reference position of the ZMP, and $\omega = \sqrt{\frac{g}{h}}$. $\mathbf{z}_d$ is computed instantaneously from the desired capture point (CP) positions and velocities, in fact:

$$\mathbf{c} = \mathbf{x} + \frac{1}{\omega}\dot{\mathbf{x}}, \tag{6a}$$

$$\xrightarrow[\frac{\partial}{\partial t}]{} \quad \dot{\mathbf{c}} = \dot{\mathbf{x}} + \frac{1}{\omega}\ddot{\mathbf{x}} = \dot{\mathbf{x}} + \omega(\mathbf{x} - \mathbf{z}),$$

$$\rightarrow \quad \mathbf{z}_d = \mathbf{c} - \frac{1}{\omega}\dot{\mathbf{c}}_d, \tag{6b}$$

with $\mathbf{c}$ and $\dot{\mathbf{c}}$ the CP and its velocity, respectively. The term $\dot{\mathbf{c}}_d$ can be treated as a new reference input based on a P feedback law plus feed-forward:

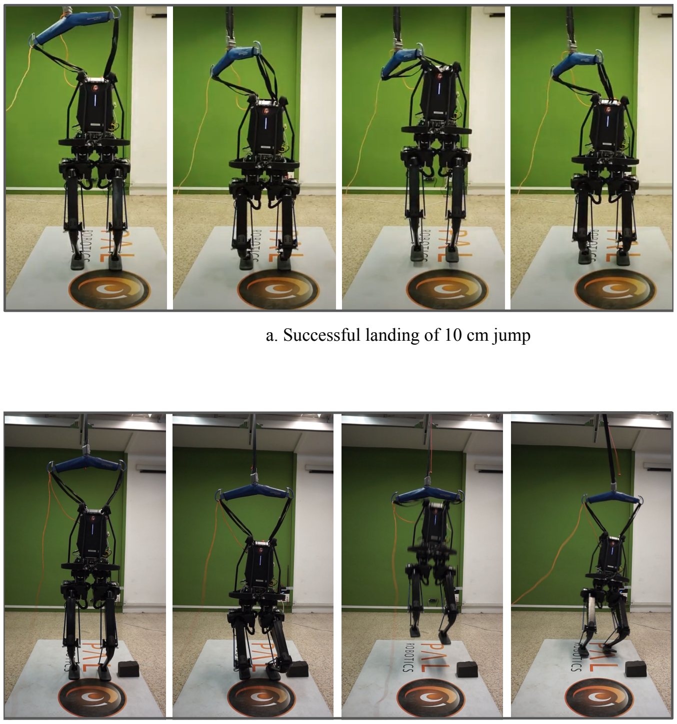
$$\dot{\mathbf{c}}_d = \dot{\mathbf{c}}_r + \mathbf{K}(\mathbf{c}_r - \mathbf{c}). \tag{7}$$

VII. Results

We tested our preliminary control architecture on the Kangaroo robot to perform a series of jumps. The robot successfully jumped at various heights several times demonstrating robustness and repeatability, even though it was not able to always land correctly without falling, especially for higher jumps where the landed COM position stays far from the center of the support polygon. Fails in landings were mostly caused by issues related to force mapping from joints to actuators and state-estimation. Figure 4 shows one of the 10 cm jumps with a correct landing, on the top sequence, and a higher jump, more than 20 cm, with a failed landing, on the bottom sequence. The jumping videos can be found at https://www.youtube.com/playlist?list=PLg5jtQ_hM1RezuPS6CCQpbiv27L_ZAdtI.

VIII. Conclusion and Future Works

In this short paper, we presented the Kangaroo platform, introducing its mechanical design, and the planning and control software pipeline. The entry point of this pipeline is the reduced model which simplifies the complex kinematic structure of the real system. Closed and parallel linkages are considered at transmission-level, when computed torques from OSID, using the reduced model, are transformed into proper force inputs for the motors. The robustness and resilience of Kangaroo have been validated through a series of experiments with the robot jumping at various heights.



a. Successful landing of 10 cm jump

b. Failure landing of more than 20 cm jump

Fig. 4: In the upper sequence Kangaroo jumping 10 cm and successfully landing. In the lower sequence a high jump with a failed landing.

In the close future, we plan is to substitute this, so-called *jumping controller*, with a model predictive controller, for example, based on the centroidal dynamics of the system. This will improve disturbances rejection, particularly for higher jumps. Future works also include to feed-forward the trajectories generated through the existing trajectory optimization framework to generate different kinds of motions such as lateral jumps, forward jumps, spins, etc. Finally, the employment of the full model for the OSID computation will also be considered to accurately compensate for the dynamics of the system.

References

[1] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[2] S. Chitta, E. Marder-Eppstein, W. Meeussen, V. Pradeep, A. R. Tsouroukdissian, J. Bohren, D. Coleman, B. Magyar, G. Raiola, M. Lüdtke *et al.*, "ros_control: A generic and simple control framework for ros," *The Journal of Open Source Software*, vol. 2, no. 20, pp. 456–456, 2017.

[3] C. Knabe, R. Griffin, J. Burton, G. Cantor-Cooke, L. Dantanarayana, G. Day, O. Ebeling-Koning, E. Hahn, M. Hopkins, J. Neal *et al.*, "Team valor's escher: A novel electromechanical biped for the darpa robotics challenge," in *The DARPA Robotics Challenge Finals: Humanoid Robots To The Rescue*. Springer, 2018, pp. 583–629.

[4] F. Ruscelli, A. Laurenzi, E. Mingo Hoffman, and N. G. Tsagarakis, "A fail-safe semi-centralized impedance controller: Validation on a parallel kinematics ankle," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

[5] N. A. Radford, P. Strawser, K. Hambuchen, J. S. Mehling, W. K. Verdeyen, A. S. Donnan, J. Holley, J. Sanchez, V. Nguyen, L. Bridgwater *et al.*, "Valkyrie: Nasa's first bipedal humanoid robot," *Journal of Field Robotics*, vol. 32, no. 3, pp. 397–419, 2015.

[6] C. Hubicki, J. Grimes, M. Jones, D. Renjewski, A. Spröwitz, A. Abate, and J. Hurst, "Atrias: Design and validation of a tether-free 3d-capable spring-mass bipedal robot," *The International Journal of Robotics Research*, vol. 35, no. 12, pp. 1497–1521, 2016.

[7] E. Guizzo, "By leaps and bounds: An exclusive look at how boston dynamics is redefining robot agility," *IEEE Spectrum*, vol. 56, no. 12, pp. 34–39, 2019.

[8] T. Apgar, P. Clary, K. Green, A. Fern, and J. W. Hurst, "Fast online trajectory optimization for the bipedal robot cassie." in *Robotics: Science and Systems*, vol. 101, 2018, p. 14.

[9] J. Hurst, "Walk this way: To be useful around people, robots need to learn how to move like we do," *IEEE Spectrum*, vol. 56, no. 3, pp. 30–51, 2019.

[10] D. Mronga, S. Kumar, and F. Kirchner, "Whole-body control of series-parallel hybrid robots," 2021.

[11] Apptronik inc., "P170 orion." [Online]. Available: https://apptronik.com/product/apptronik-p170-orion/

[12] V. Barasuol, O. A. Villarreal-Magaña, D. Sangiah, M. Frigerio, M. Baker, R. Morgan, G. A. Medrano-Cerda, D. G. Caldwell, and C. Semini, "Highly-integrated hydraulic smart actuators and smart manifolds for high-bandwidth force control," *Frontiers in Robotics and AI*, vol. 5, 2018.

[13] J. Eβer, S. Kumar, H. Peters, V. Bargsten, J. d. G. Fernandez, C. Mastalli, O. Stasse, and F. Kirchner, "Design, analysis and control of the series-parallel hybrid rh5 humanoid robot," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2021, pp. 400–407.

[14] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.